HHaaHMM: A Hyper-Heuristic as a Hidden Markov Model

Willem Van Onsem, Bart Demoen, and Patrick De Causmaecker

KU Leuven, Department of Computer Science

Abstract. A new hyper-heuristic algorithm is presented. The approach uses an expectation-maximization strategy in order to find the most likely compressed representation of the hyper-heuristic process. The approach is in principle to calculate a good heuristic at any point in time given the old data.

1 Approach

Our approach is based on a hidden Markov model (HMM). We first give a definition of a hidden Markov model together with its relation to hyper-heuristics. Next we describe how some problems were addressed to make this approach feasible.

1.1 Input-Output Hidden Markov Models (IOHMMs) and Hyper-heuristics

A hidden Markov model is a mathematical formalism that is defined formally as follows:

Definition 1 (Hidden Markov model). *A hidden Markov model λ with n hidden states and m output characters is a tuple $\lambda = \langle \pi, A, B \rangle$ with π a n-vector, A an $n \times n$ matrix and B an $n \times m$ matrix. π describes the initial probability distribution over n hidden states, A is a matrix that describes the transition of hidden states after each observation, and B describes the the probability b_{ij} given the model is in state i to make observation j.*

Hidden Markov models are trained with a list of observations $\boldsymbol{o}$ using the *Baum-Welch algorithm*[1970]. This algorithm aims to maximize the likelihood of the seen observations by using an *expectation-maximization* approach but is constrained by a limited number of parameters to set. *Bause*[2003] generalized the concept by defining *Input-Output Hidden Markov models* (IOHMM): Hidden Markov models where the next state s_j and the output character y_i depend on the state and an input symbol x_i from a finite alphabet X.

A hyper-heuristic has much in common with an IOHMM. The active solutions correspond to the hidden states, since hyper-heuristics have no knowledge of these solutions. The output states correspond to what is visible: the fitness value and the time taken to transform one solution into another one. Finally the input symbols are the heuristics applied to the solutions. Figure 1 illustrates this principle.

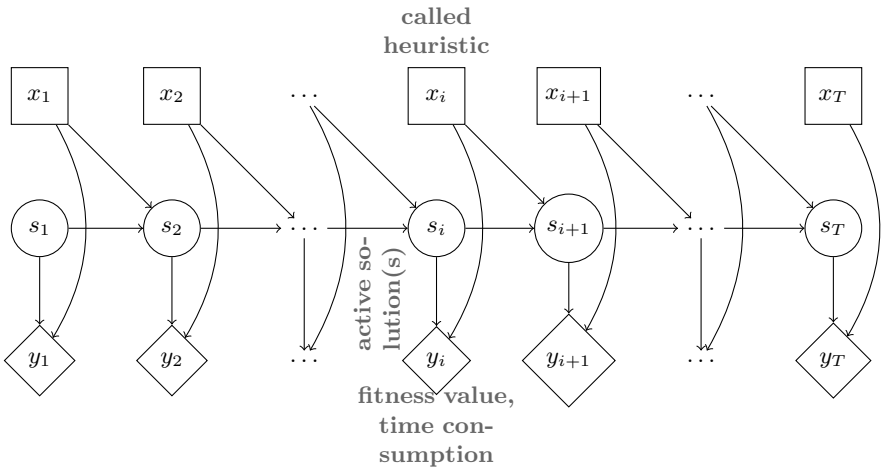

Fig. 1. The correspondence between an IOHMM and a hyper-heuristic.

One can however not translate a hyper-heuristic directly into a IOHMM. The basic versions of an IOHMM work with a finite input- and output-alphabet X and Y. In the hyper-heuristic context it is likely that the input and output are continuous.

Furthermore the hidden states should correspond to all possible sets of solutions. This would result in a super-exponential amount of hidden states. Since the number of interactions and observations a hyper-heuristic can perform in a reasonable amount of time is limited, the algorithm can never learn a significant amount of the hidden states.

It is however reasonable to assume that the behavior of the hyper-heuristic process can be learned with only a limited number of hidden states. One is for instance interested whether a heuristic will improve an existing solution. The exact solution is not relevant and unknown. It is likely that with a very limited number of hidden states, the progress of a hyper-heuristic can be described accurately. We need to discretize the input- and output-alphabets X and Y and find reasonable parameters for the number of hidden states. We describe these in the following section.

1.2 Discretization

Input alphabet Some heuristics use the *depth of search* and *intensity of mutation* parameters. Therefore the number of possible inputs is infinite. One can however argue that a small difference to one of the heuristic parameters won't result in a significant different resulting solution. In our implementation, we only used the type of heuristic as the input value.

It is our intention to extend this approach. By using a clustering algorithm, we can divide the input space into m regions. As more data becomes available the number of clusters can be increased. Currently however there is no straightforward way to map clusters back on heuristics and furthermore we need to design a well suited supervised clustering algorithm.

Output alphabet The possible outcomes of applying a heuristic to a solution is continue: both the difference in fitness value and the time consumed are continuous parameters. *Van Onsem, Demoen and De Causmaecker*[2014] argue however that the difference in fitness value is not always a reliable metric: say you multiply the fitness value with a constant, a hyper-heuristic should be oblivious to this fact. Therefore we limited the output alphabet to five values: *improve*, *worse*, *worse-time*, *same*, and *same-time*. These values are ordered from best to worst. It may seem controversial that a worse solution is assumed to be better than generating the same solution. If however the same solution is generated, the computational effort is wasted while a worse solution sometimes gets a system out of a local optimum.

Hidden states The number of hidden states is an unknown parameter. As the number of hidden states increases, we expect that the model will be able to describe the process better. On the other hand if too much hidden states are taken into account, the algorithm can't learn that model and certainly not generalize the observed data.

We can however increase the number of hidden states as more data becomes available. The number of parameters related to the an IOHMM with m input symbols, n hidden states and o output symbols scales with $\mathcal{O}(m \cdot n \cdot (n + o))$. Based on this fact, we let the number of hidden states scale with $n = \mathcal{O}\left(\sqrt{\log t}\right)$ with t the number of observations. If the number of hidden states is incremented, the IOHMM is not reset. The hidden state that is the least determined in output character and next state is split into two independent states in order to address this uncertainty in the model.

1.3 Determining the next input

Even if we train a IOHMM appropriately, it is still an open question which input we should take next. We can of course calculate the next input character based on

the desired output character. However this can imply that we get stuck the next state and eventually won't make any progress. *Wissner-Gross and Freer*[2013] argue that intelligent systems aim to maximize the entropy. In this case, this would imply that we tend to make the distribution over the next hidden states as uniform as possible. We aim to implement both strategies in our approach: the next input value is determined by maximizing immediate profit together with the entropy of the future hidden state distribution.

2 Inductive bias

The implemented model is oblivious to any sample data. Therefore it is unlikely that the algorithm will outperform tailor made hyper-heuristics. We are merely interested in the result of such an approach compared to more tweaked approaches. The advantage of the presented approach however is that the algorithm has almost no inductive bias: it can learn almost any kind of rules. For instance with this approach, the hyper-heuristic should be able to learn:

1. That some heuristics are idempotent: applying a heuristic a second time consecutively has no effect.
2. That local search heuristics can only improve solutions (or return the same one).
3. That repeatedly applying the same type of heuristic decreases the effectiveness of the heuristic.

3 Conclusion

Our aim of this hyper-heuristic is not to outperform other hyper-heuristics: it is likely that other hyper-heuristics are more trained on the set of problems that occur in the test bench. Our theoretical model has a direct link with what a hyper-heuristic basically does. Discretizing the input and output alphabet reasonably still remains an open problem. An advantage of a IOHMM is that it has nearly no inductive bias and thus can learn several rules on the long term.

References

[1970] L. E. Baum, T. Petrie, G. Soules, and N. Weiss,*A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains*, Ann. Math. Statist., vol. 41, no. 1, pp. 164171, 1970.

[2003] Flako Bause. Input-output hidden markov models for aggregation of performance models. Technical report SFB559-03010, Dortmund (2003)

[2013] A. D. Wissner-Gross and C. E. Freer. *Causal Entropic Forces* Phys. Rev. Lett. 110, 168702, Published 19 April 2013.

[2014] W. Van Onsem, B. Demoen, and P. De Causmaecker. Hyper-criticism: A critical reflection on todays hyper-heuristics. In *Proceedings of the 28th Annual Conference of the Operational Research Society*, volume 28, Mons, Belgium, January 2014. ORBEL.